Value-based approach for reinforcement learning

先端人工知能論2 2017年11月21日

株式会社Preferred Networks
前田 新一
片岡 俊基

ichi@preferred.jp

自己紹介

前田 新一

- 2004年 奈良先端科学技術大学院大学 博士号取得→助手
  学習する機械に興味を持ち、ベイズ推定に基づく
  機械学習全般 と 強化学習の理論研究

- 2008年 京都大学情報学研究科 助教
  神経画像解析, X線CT, 深層学習と強化学習のアルゴリズム研究

- 2017年 Preferred Networks, Researcher

Preferred Networks

講義パート内容

- Q-learning 5min
- Deep Q Network (DQN) 10min
- Variants of DQN 4min x 5 = 20min
  Double DQN
  Dueling Networks
  Prioritized Experience Replay
  Multi-step learning
  Distributional RL
- Extension to Continuous action 10min
  (DDPG), NAF, input convex NN

価値ベース(DQN関連)の最近の
強化学習の研究について説明します

Preferred Networks

強化学習のおさらい

目的

方策πが定まったもとで
長期的な報酬(累積報酬)の予測

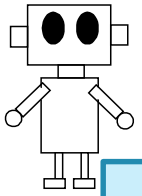
$$V^{\pi}(s_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots \mid s_t \right]$$

E[・]は期待値(expectation)の意味。

$$\prod_{k=t}^{\infty} (p(r_{k+1}|s_{k+1}, s_k, a_k) p(s_{k+1}|s_k, a_k) p_{\pi}(a_k|s_k))$$

γは $0 \leq \gamma < 1$ を満たす定数

方策(行動則)
$\pi : p_{\pi}(a_t \mid s_t)$
行動 a_t



s_t

エージェント

環境(外界)

状態遷移確率
$s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$

報酬確率 (報酬関数)
$r_{t+1} \sim p(r_{t+1} \mid s_t, a_t, s_{t+1})$

観測される状態 s_{t+1}
観測される報酬 r_{t+1}
(or 知覚する報酬)

Preferred Networks

強化学習のおさらい

目的

方策πが定まったもとで
長期的な報酬(累積報酬)の予測

累積報酬を最大にするための行動則の獲得

$$\pi^* = \arg\max_{\pi} V^{\pi}(s_t)$$

$$V^*(s) := V^{\pi*}(s)$$

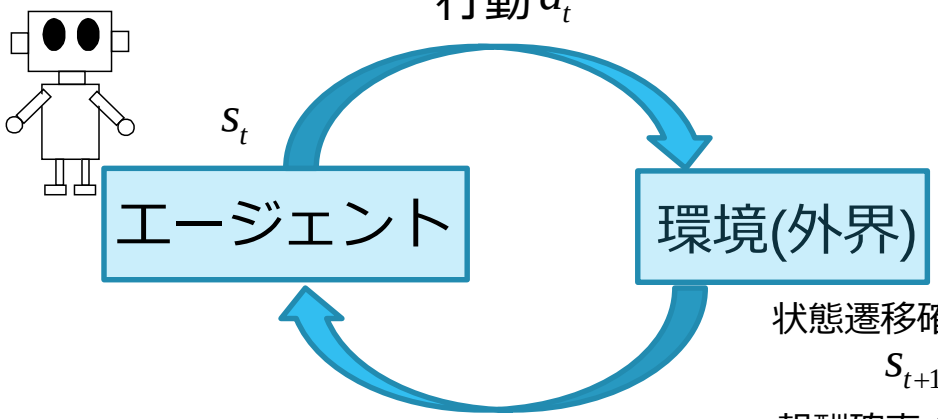
$$V^{\pi}(s_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots \mid s_t \right]$$

E[・]は期待値(expectation)の意味。

$$\prod_{k=t}^{\infty} (p(r_{k+1}|s_{k+1}, s_k, a_k) p(s_{k+1}|s_k, a_k) p_{\pi}(a_k|s_k))$$

γは $0 \leq \gamma < 1$ を満たす定数

方策(行動則)
$\pi : p_{\pi}(a_t \mid s_t)$



状態遷移確率

$$s_{t+1} \sim p(s_{t+1} \mid s_t, a_t)$$

報酬確率 (報酬関数)

$$r_{t+1} \sim p(r_{t+1} \mid s_t, a_t, s_{t+1})$$

Preferred Networks

ベルマン方程式

価値関数に成り立つ再帰的な関係式

$$V^{\pi}(s_t) = E^{\pi}\left[r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t \right]$$

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma Q^{\pi}(s_{t+1}, a_{t+1}) \mid s_t \right]$$

最適ベルマン方程式 (Bellman Optimality Equation)

$$\begin{aligned} V^*(s_t) &= E^{\pi*}[r_{t+1} + \gamma V^*(s_{t+1}) | s_t] \\ &= \int p_{\pi*}(a_t|s_t) E[r_{t+1} + \gamma V^*(s_{t+1}) | s_t, a_t] da_t \\ &= \max_{a_t} E[r_{t+1} + \gamma V^*(s_{t+1}) | s_t, a_t] \\ &= \max_{a_t} Q^*(s_t, a_t) \end{aligned}$$

$$Q^*(s_t, a_t) = E\left[r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \mid s_t, a_t \right]$$

連続時間でのベルマン方程式
ハミルトン・ヤコビ・ベルマン方程式
Hamilton-Jacobi-Bellman equation (HJB equation)

$$V^*(s(t)) = \max_a \left\{ \int_t^{t+dt} r(s(k), a(k)) dk + V^*(s(t+dt)) \right\}$$

Preferred Networks

最適価値関数をどのようにして求めるか？

ベルマンオペレータはsupノルムに対して縮小写像なので
ベルマン方程式を満たす価値関数は、唯一
したがって、最適ベルマン方程式を満たす価値関数は最適価値関数

$$f(s_t) = \max_{a_t} E\left[r_{t+1} + \gamma f(s_{t+1}) \mid s_t, a_t\right] \quad {}^{\forall} s_t \in S$$

$$\Rightarrow \quad f(s_t) = V^*(s_t) \quad {}^{\forall} s_t \in S$$

$$f(s_t, a_t) = E\left[r_{t+1} + \gamma \max_{a_{t+1}} f(s_{t+1}, a_{t+1}) \mid s_t, a_t\right] \quad {}^{\forall}(s_t, a_t) \in S \times A$$

$$\Rightarrow \quad f(s_t, a_t) = Q^*(s_t, a_t) \quad {}^{\forall}(s_t, a_t) \in S \times A$$

最適ベルマン方程式を満たす価値関数を求める

Preferred Networks

最適価値関数をどのようにして求めるか？

最適ベルマンオペレータによる更新を繰り返せばよい

しかし、すべての状態に対して、期待値を計算して更新するのは困難

$$V(s_t) = \max_{a_t} E\left[r_{t+1} + \gamma V(s_{t+1}) \mid s_t, a_t\right] \quad {}^{\forall} s_t \in S$$

$$Q(s_t, a_t) = E\left[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \mid s_t, a_t\right] \quad {}^{\forall}(s_t, a_t) \in S \times A$$

サンプル平均による期待値の近似

深層学習を用いた強化学習

Preferred Networks

関数近似を用いたときの収束の有無

表 2.3　SARSA, Q 学習による行動価値関数 Q^{π}, Q^{*} の近似

手法	方策	関数近似器	収束の保証
SARSA	方策オン型	線形	滑らかな方策反復であり
SARSA	方策オン型	非線形	なし
Q 学習	方策オフ型	線形	方策固定であり
Q 学習	方策オフ型	非線形	なし

収束の保証がないことは以前から問題視されていた

NNを関数近似器に使っていた研究もあったが、安定性がなく
性能を発揮するためには特徴量の作り込みが重要と考えられた

Preferred Networks

Deep Q-Network (DQN) (Mnih+ 14, Mnih+ 15)

学習アルゴリズムの工夫した点

- Q 関数を表わすネットワーク構造
- Fitted Q (ターゲットを固定し安定性改善)
- 経験再生 (Experience Replay)
  (学習に用いるサンプルの偏り抑制)
- 報酬のクリッピング [-1,1]
- 探索方策ε-greedyのεを1.0から0.1に減衰

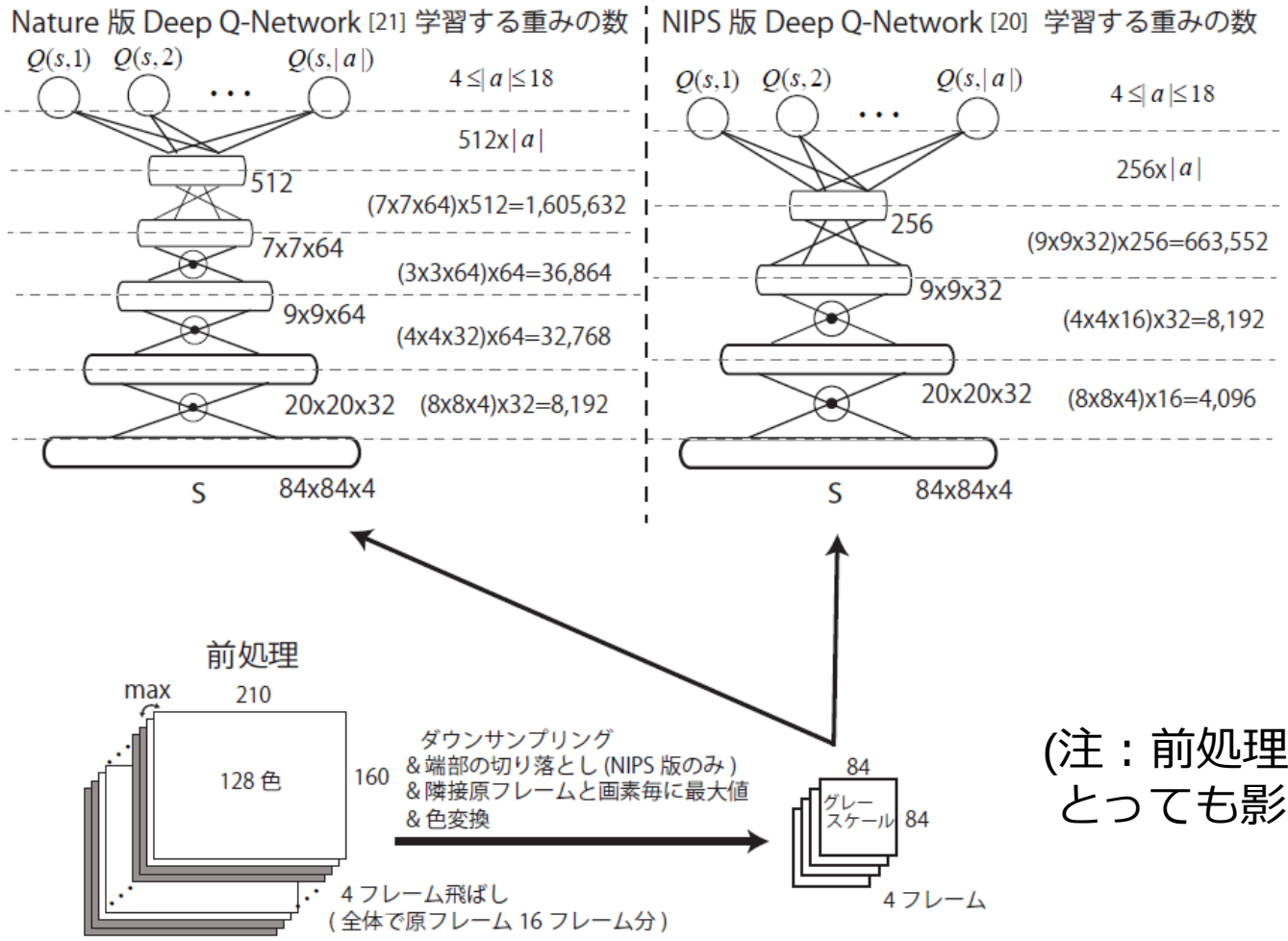
Ms. Pack-Man

Space Invaders

Preferred Networks

ネットワーク構造

行動の数だけ出力Q(s,a)をもつネットワーク



(注：前処理は学習にとっても影響します)

OpenAIのgymのAtariとArcade Learning EnvironmentのAtariは別物

Preferred Networks

Neural Fitted Q (Riedmiller, 2005)

損失関数

$$J(\theta) = E\left[(y_t - Q(s_t, a_t; \theta))^2 \right]$$

$$y_t = r_{t+1} + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta')$$

定期的にθ'を現在のθに置き換え
(しばらくはターゲットを固定)

Ms. Pack-Man

Space Invaders

教師あり学習的な損失関数に変更することで安定化

Preferred Networks

経験再生 (Experience replay) (Lin, 1993)

TD誤差を連続する時刻間で取得
→ サンプル間の相関が高く、バイアス発生、収束性悪化

損失関数

$$J(\theta) = E\left[(y_t - Q(s_t, a_t; \theta))^2 \right]$$

$$y_t = r_{t+1} + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta')$$

サンプルの再利用

$$\nabla J(\theta) = 2E\left[(y_t - Q(s_t, a_t; \theta)) \nabla Q(s_t, a_t; \theta) \right]$$

(誤差$\left[(y_t - Q(s_t, a_t; \theta)) \nabla Q(s_t, a_t; \theta) \right]$のクリッピングも安定化に寄与と報告)

状態遷移セット(s,a,s',r)を直近の過去のデータ(replay memory)からランダムに選択し、TD誤差 $y_t - Q(s_t, a_t; \theta)$ を計算

Preferred Networks

Replay memory 直近100万フレーム

ターゲットQの固定と経験再生の効果

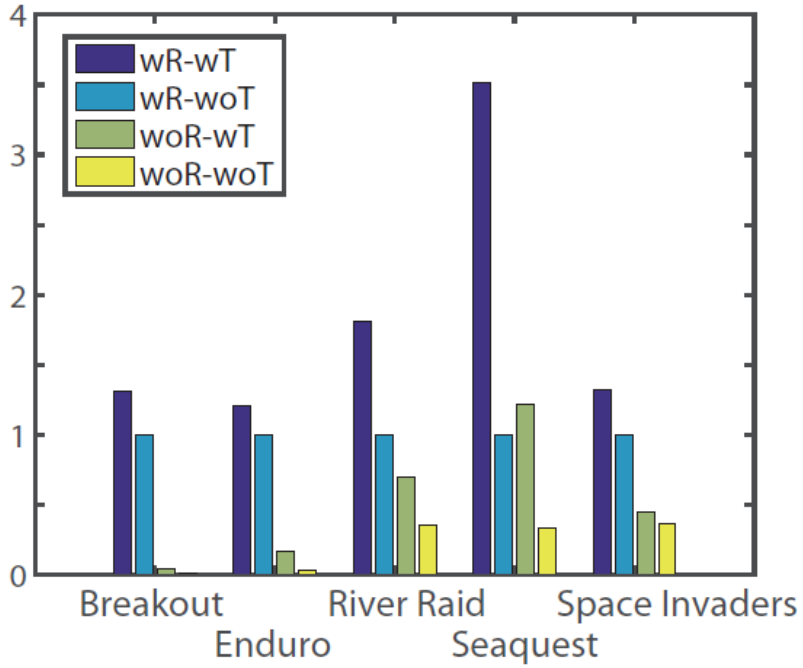
WT：ターゲットQの固定あり

WoT：ターゲットQの固定なし

WR：経験再生あり

WoR：経験再生なし

wR-woTを1としたときの
パフォーマンス



Preferred Networks

Variants of DQN

- Double DQN (van Hasselt+ 16)
- Dueling Networks (Wang+15)
- Prioritized Experience Replay (Schaul+ 16)
- Multi-step learning (Sutton 98)
- Distributional RL (Bellemare+ 17, Dabney+ 17)

Double DQN (van Hasselt+ 16)

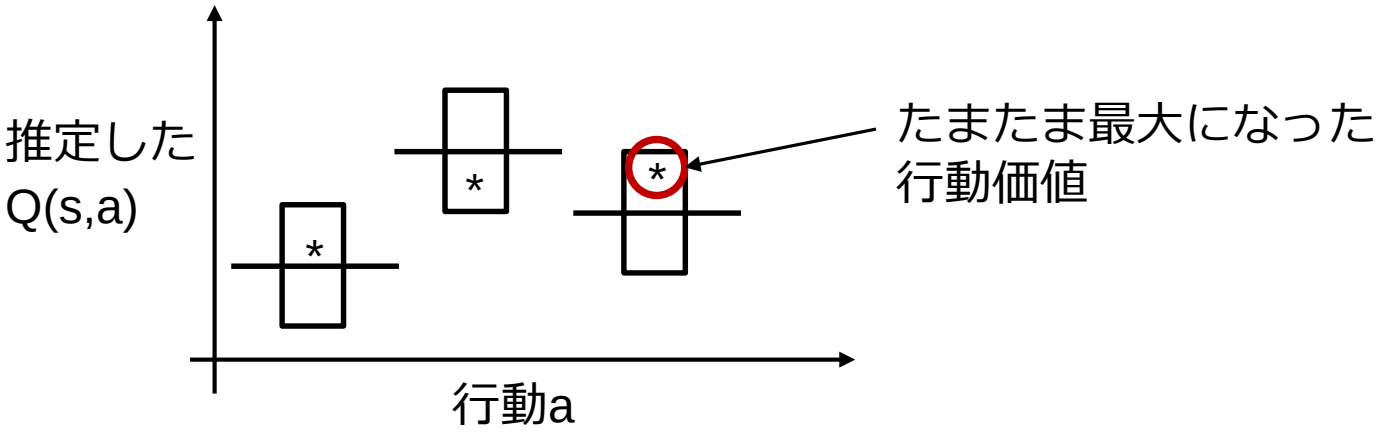
推定値の最大を選ぶことによるバイアス

$$Q(s, a = i; \theta_t) = Q^*(s, a = i) + \epsilon_i$$

推定誤差ϵ_iが平均ゼロの確率変数とすると、

$$\max_i Q(s, a = i; \theta_t) = \max_i (Q^*(s, a = i) + \epsilon_i)$$

Qの推定値の最大を選ぶことによるバイアスが発生



Preferred Networks

Double DQN (van Hasselt+ 16)

Fitted Qのターゲットの変更

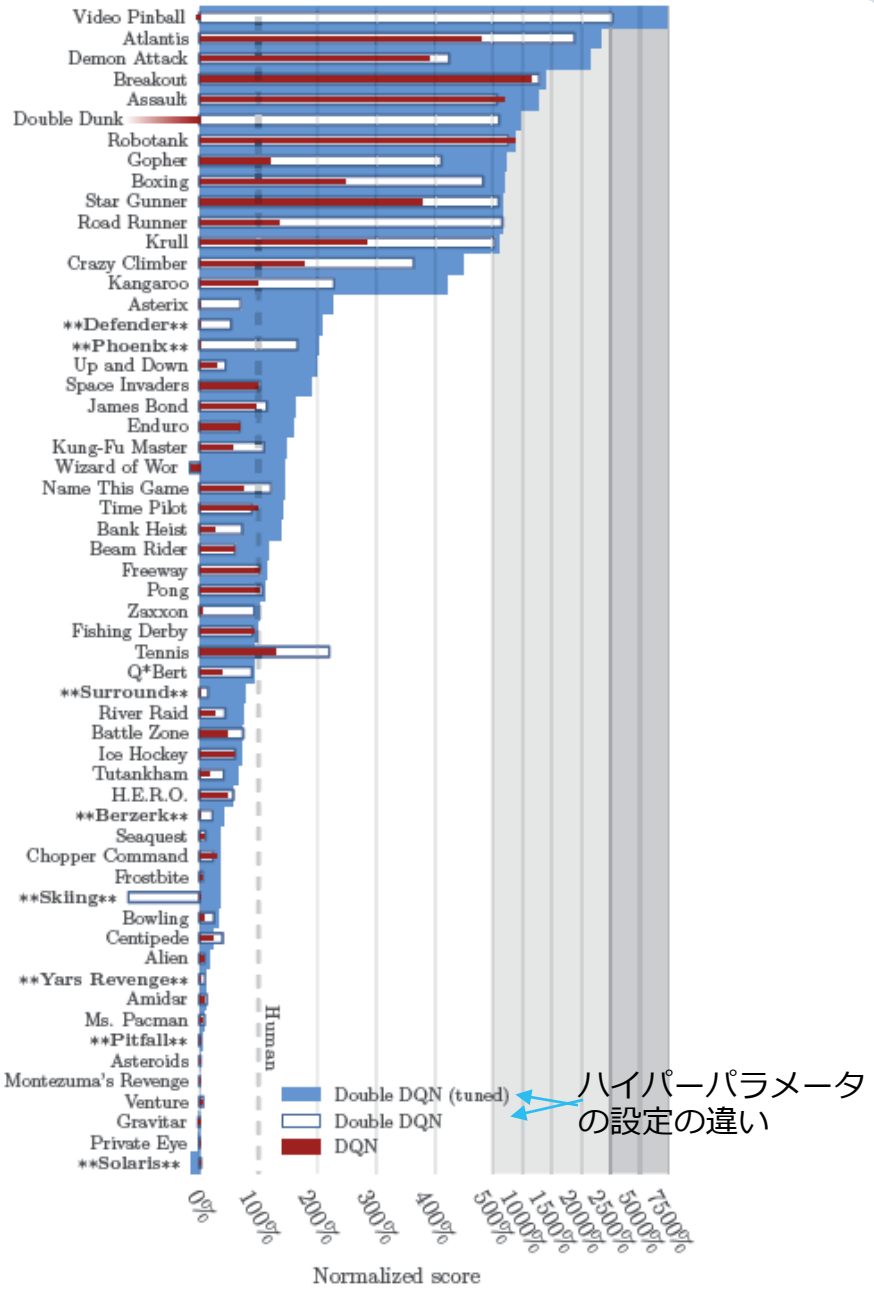
$$y_t = r_{t+1} + Q(s_{t+1}, \hat{a}_{t+1}; \theta_t)$$

where

$$\hat{a}_{t+1} = \arg\max_a Q(s_{t+1}, a; \theta_t)$$

$$\hat{a}_{t+1} = \arg\max_a Q(s_{t+1}, a; \theta_t^-)$$

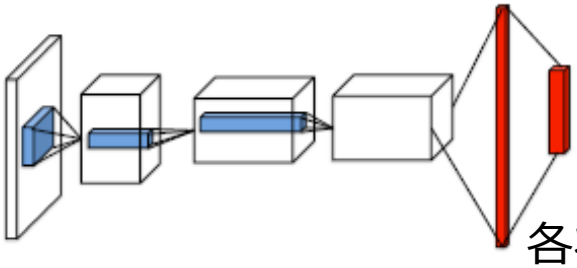
θ_tとθ_t^-を入れ替えて交互に学習



Preferred Networks

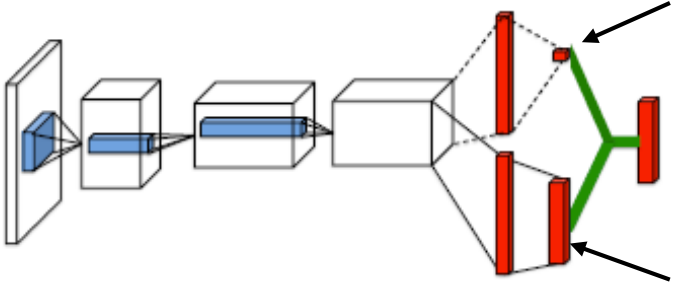
Dueling Networks (Wang+15)

通常のQネットワーク



各状態に対して1つの出力V

Dueling Qネットワーク



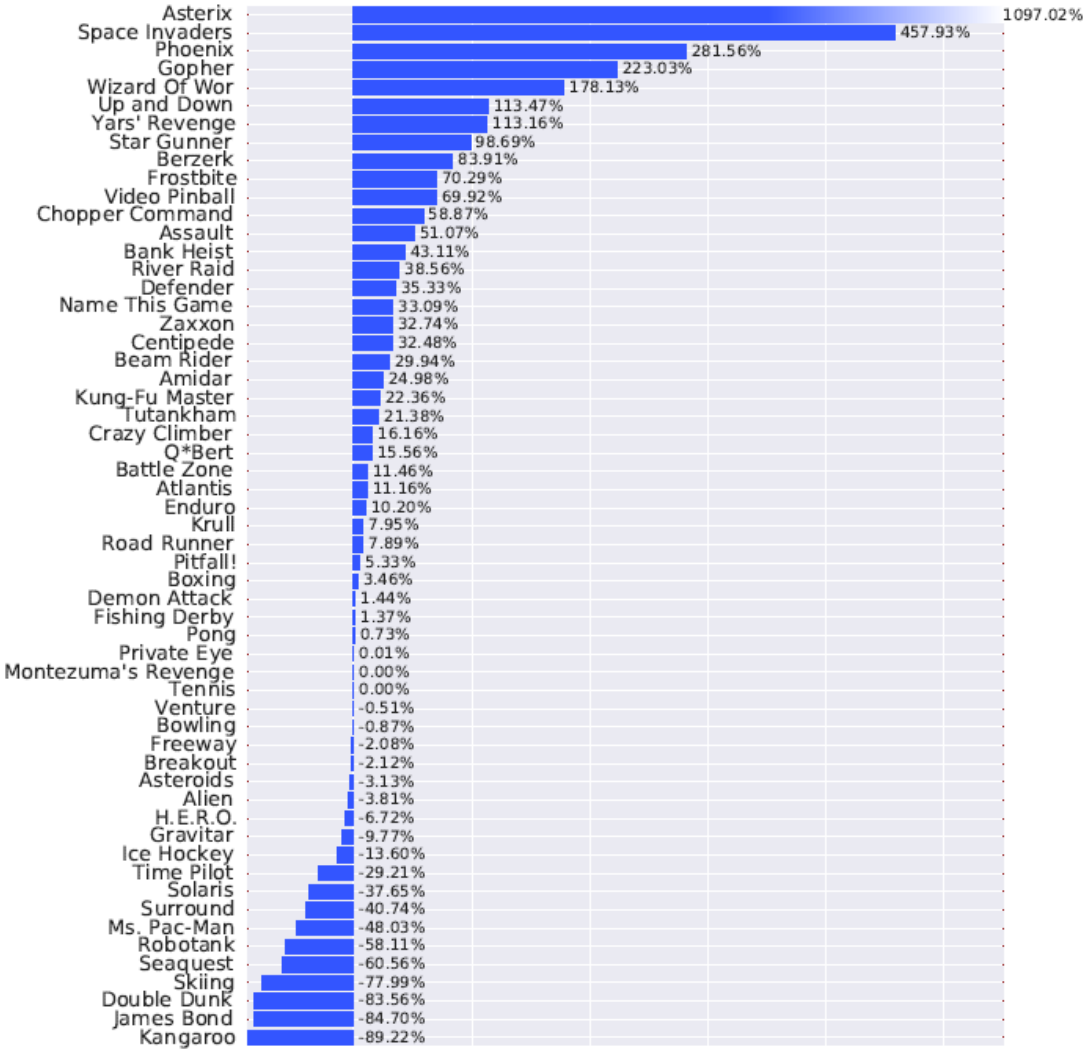
各状態に対してアクションごとの出力A

$$Q(s,a) = V(s) + \left(A(s,a) - \frac{1}{|\mathrm{A}|} \sum_{a'} A(s,a') \right)$$

Vはどの行動を選択した場合でも学習
行動に依存しない状態価値を学習可能

Preferred Networks

Dueling Networks (Wang+15)



多くのゲーム
で大きく性能向上

Preferred Networks

Prioritized Experience Replay (Schaul+ 16)

TD誤差δの選び方を変えてみる

Replay Memoryからの状態遷移セット(s,a,s',r)を
TD誤差の大きいものから優先的に選択

間違いやすい項目を頻繁に出現させて覚える暗記法のような感じ

$$\delta_t = r_{t+1} + \max_{a_{t+1}} Q(s_{t+1}, \hat{a}_{t+1}; \theta') - Q(s_t, a_t; \theta)$$

$$\Delta\theta \propto E_p\left[\delta_t \nabla Q(s_t, a_t; \theta)\right]$$

p:Replay memory からの一様サンプル

$$\Delta\theta \propto E_q\left[\hat{w}\delta_t \nabla Q(s_t, a_t; \theta)\right]$$

q:サンプラー $q_i \propto (|\delta_i| + c)^\alpha$

or

$$q_i \propto \frac{1}{(\text{rank } i)^\alpha}$$

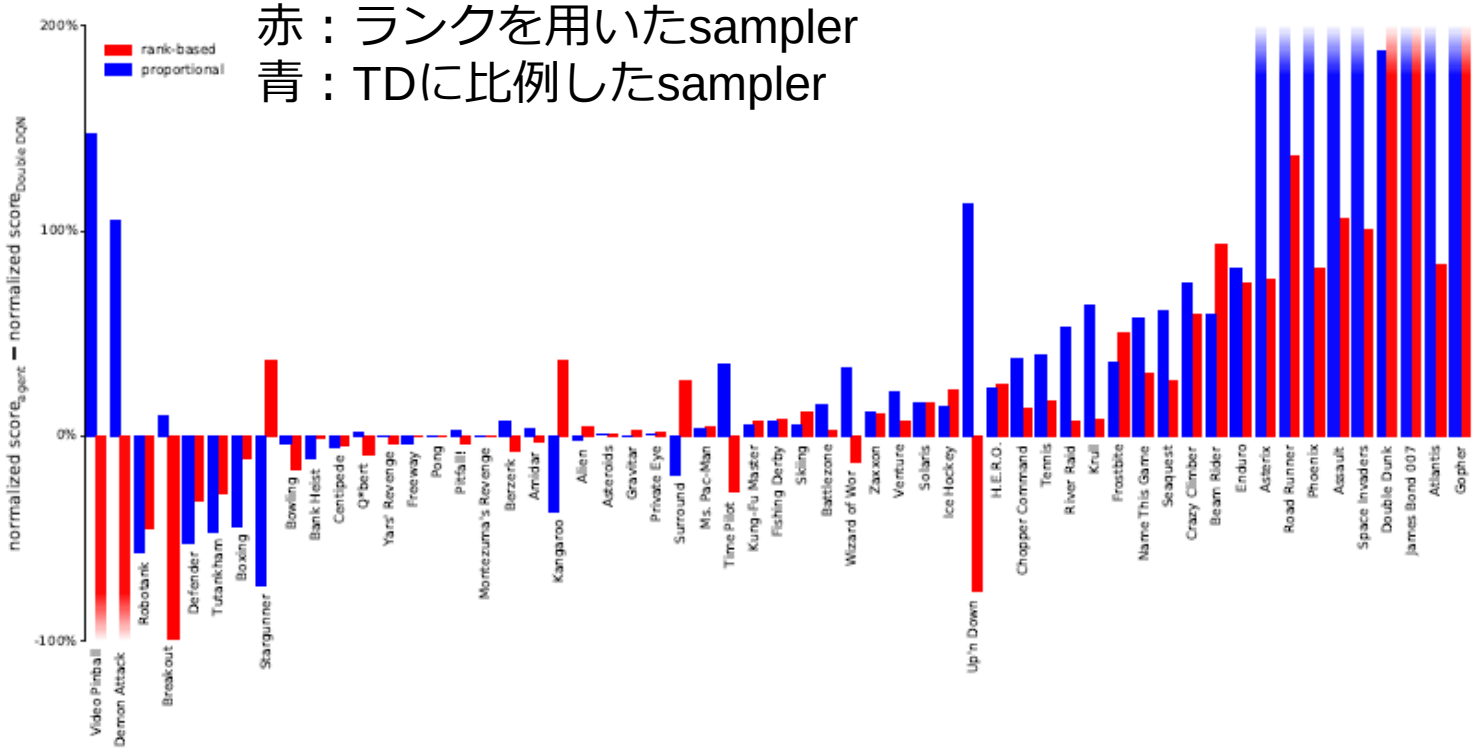
$$w = \left(\frac{p}{q}\right)^\beta = \frac{1}{(Nq)^\beta} \qquad \hat{w}_i = \frac{w_i}{\max_j w_j}$$

Preferred Networks

α, βはBias-Varianceトレードオフをコントロールするパラメータ

Prioritize Experience Replay (Schaul+ 16)

Double DQN
と比較した性能



多くのゲームで性能向上

Preferred Networks

Multi step Learning (Sutton 98)

価値関数に成り立つ再帰的な関係式

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots \mid s_t, a_t \right]$$

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma \underbrace{\left(r_{t+2} + \gamma r_{t+3} + \gamma^2 r_{t+4} + \cdots \right)} \mid s_t, a_t \right]$$

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma Q^{\pi}(s_{t+1}, a_{t+1}) \mid s_t \right]$$

誤差 $r_{t+1} + \gamma Q^{\pi}(s_{t+1}, a_{t+1}) - Q^{\pi}(s_t, a_t)$

価値関数に成り立つ再帰的な関係式を書き換える

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots \mid s_t, a_t \right]$$

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n \underbrace{\left(r_{t+n+1} + \gamma r_{t+n+2} + \cdots \right)} \mid s_t, a_t \right]$$

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n Q^{\pi}(s_{t+n+1}, a_{t+n+1}) \mid s_t \right]$$

誤差 $r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n Q^{\pi}(s_{t+n+1}, a_{t+n+1}) - Q^{\pi}(s_t, a_t)$

n→∞でモンテカルロ法(累積報酬は不偏推定)　n→1で通常のTD学習(確率要素は1ステップ)

中間のnはMulti step Learning

nを調整することでBias-Varianceトレードオフを最適化できる

Preferred Networks

Distributional RL (Bellemare+ 17, Dabney+ 17)

通常のベルマン方程式は累積報酬の期待値に関する再帰方程式

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots | s_t, a_t \right]$$

累積報酬の期待値以外にも、累積報酬の分散、
ひいては累積報酬の分布に関する再帰方程式を定義可能

$$Z(s_t, a_t) \overset{D}{=} \underbrace{r_{t+1} + \gamma Z(s_{t+1}, a_{t+1})}_{TZ(s_{t+1}, a_{t+1})}$$

分布ベルマンオペレーター TZを定義可能
Tは分布間のWasserstein距離のsupノルムに関して縮小写像

Wasserstein距離 $d_p(X, Y) \equiv \inf_{q \in \Gamma(p_X, p_Y)} \int |x - y|^p \, q(x, y) dx dy$

ここで、$\Gamma(p_X, p_Y)$ は周辺分布が $p_X(x), p_Y(y)$ となる同時分布q(x,y)の集合

$$\bar{d}_p(TZ, Z) \equiv \sup_{s,a} d_p(TZ(s,a), Z(s,a))$$

↓10月27日

サンプルから計算するのは容易ではないが(Dabney+ 17)で解決法提示

Preferred Networks

Distributional RL (Bellemare+ 17, Dabney+ 17)

通常のベルマン方程式は累積報酬の期待値に関する再帰方程式

$$Q^{\pi}(s_t, a_t) = E^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots \mid s_t, a_t \right]$$

累積報酬の期待値以外にも、累積報酬の分散、
ひいては累積報酬の分布に関する再帰方程式を定義可能

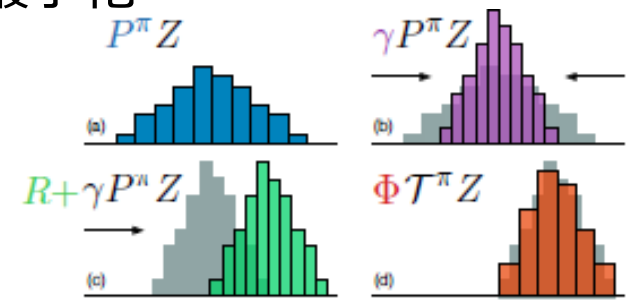
$$Z(s_t, a_t) \overset{D}{=} \underbrace{r_{t+1} + \gamma Z(s_{t+1}, a_{t+1})}_{TZ(s_{t+1}, a_{t+1})}$$

分布ベルマンオペレーター TZを定義可能
分布間のKL距離やtotal variation距離に対しては縮小写像にならない

が、サンプルから計算しやすいKL距離を最小化

$$\min_{\theta} D_{KL}(TZ, Z_{\theta})$$

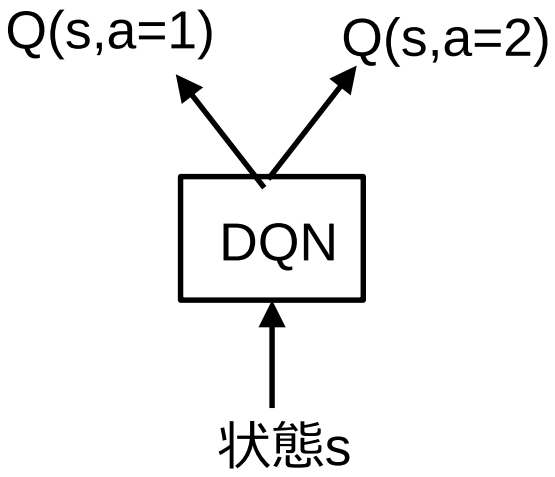
Φ はモデル分布族
(カテゴリカル分布)への射影



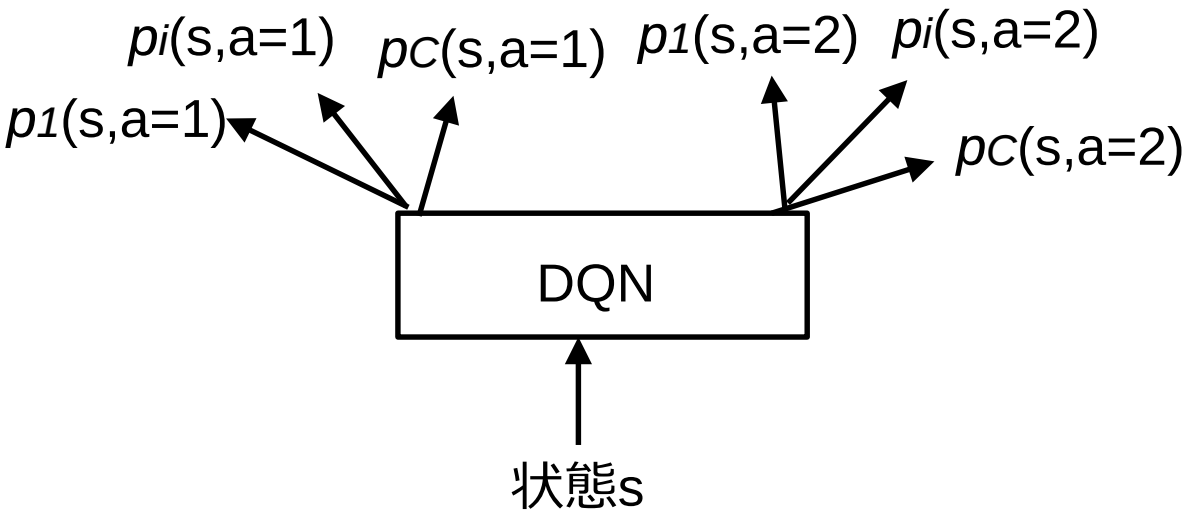
Preferred Networks

Distributional RL (Bellemare+ 17, Dabney+ 17)

通常のDQN



カテゴリカル分布を表現するDQN

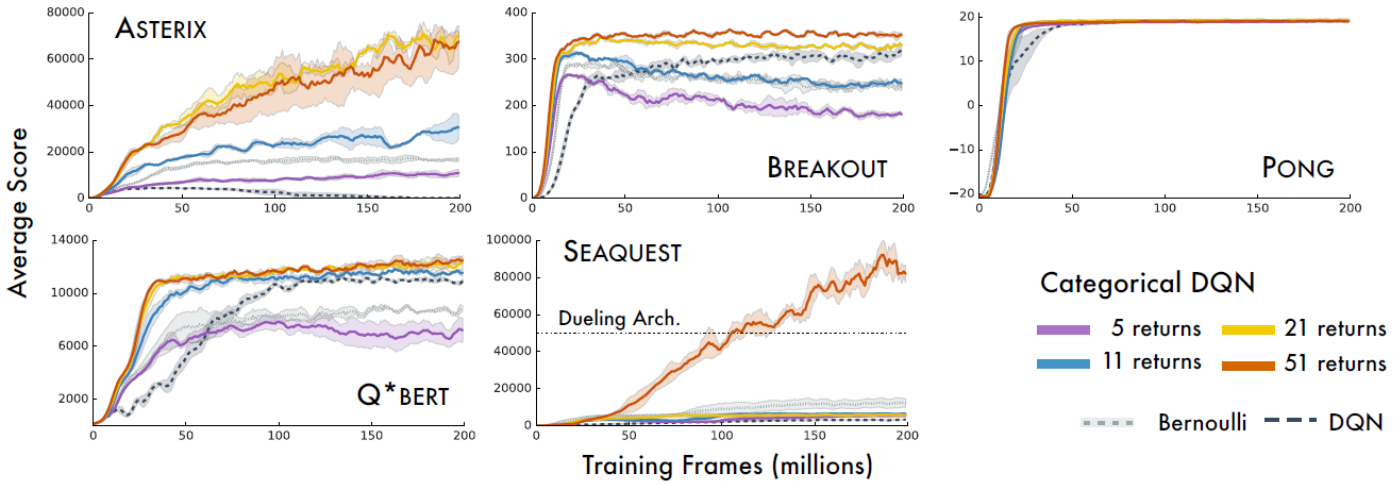


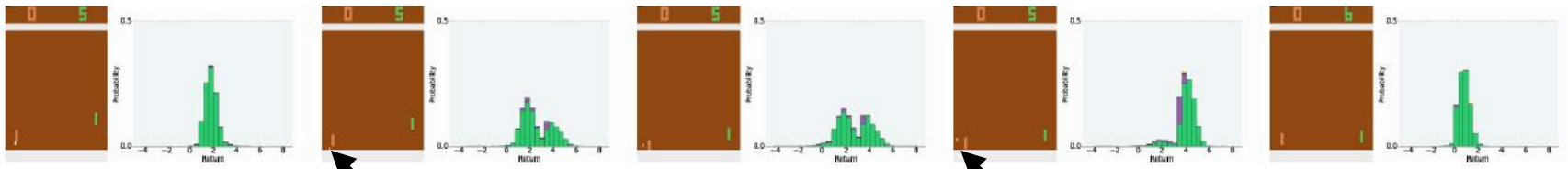
$$p_i(s,a)=p((i-1)x \leqq R<ix|s,a)$$

Preferred Networks

Distributional RL (Bellemare+ 17, Dabney+ 17)

Cはカテゴリ(短冊)の数





相手が弾を撃ち返せたか
どうかが曖昧

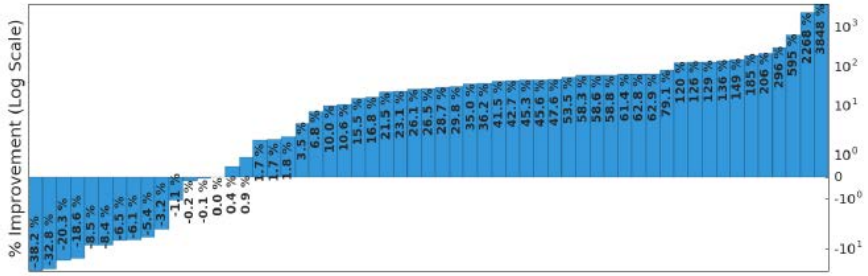
相手は弾を撃ち返し
そこねたはず

Preferred Networks

Distributional RL (Bellemare+ 17, Dabney+ 17)

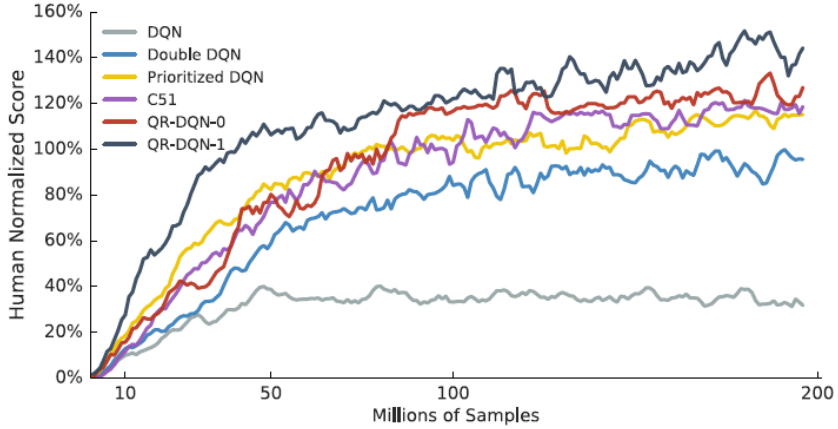
Double DQNと
比較した性能

Bellemare+ 17



ヒトと比較した性能

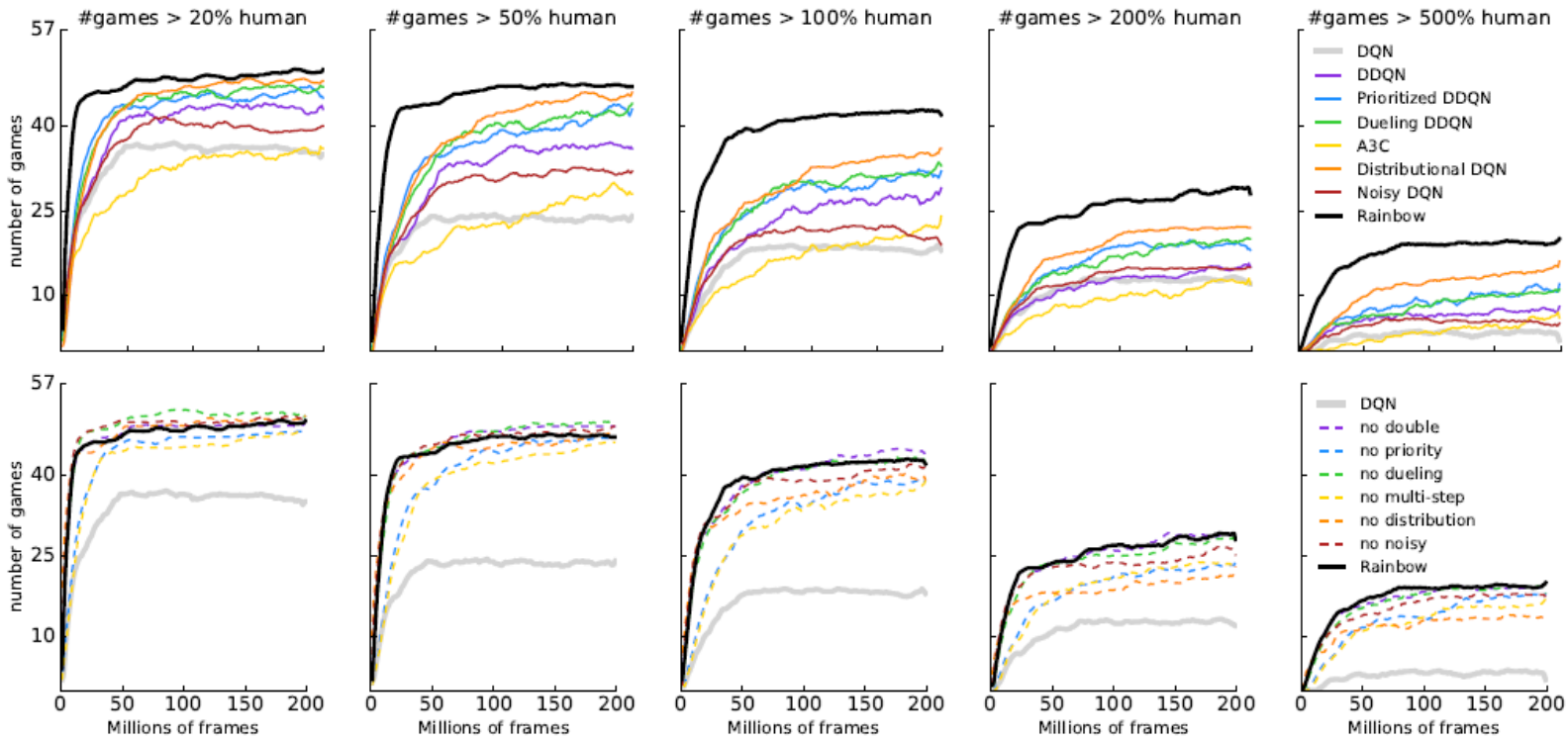
Dabney+ 17



Preferred Networks

↓10月6日

どの技術が重要なのか？　比較実験(Hessel+, 17)



↑目盛り注意 Million

分布版への変更

Prioritized experience replay

$$\delta_t = KL[Tp'|p_t]$$

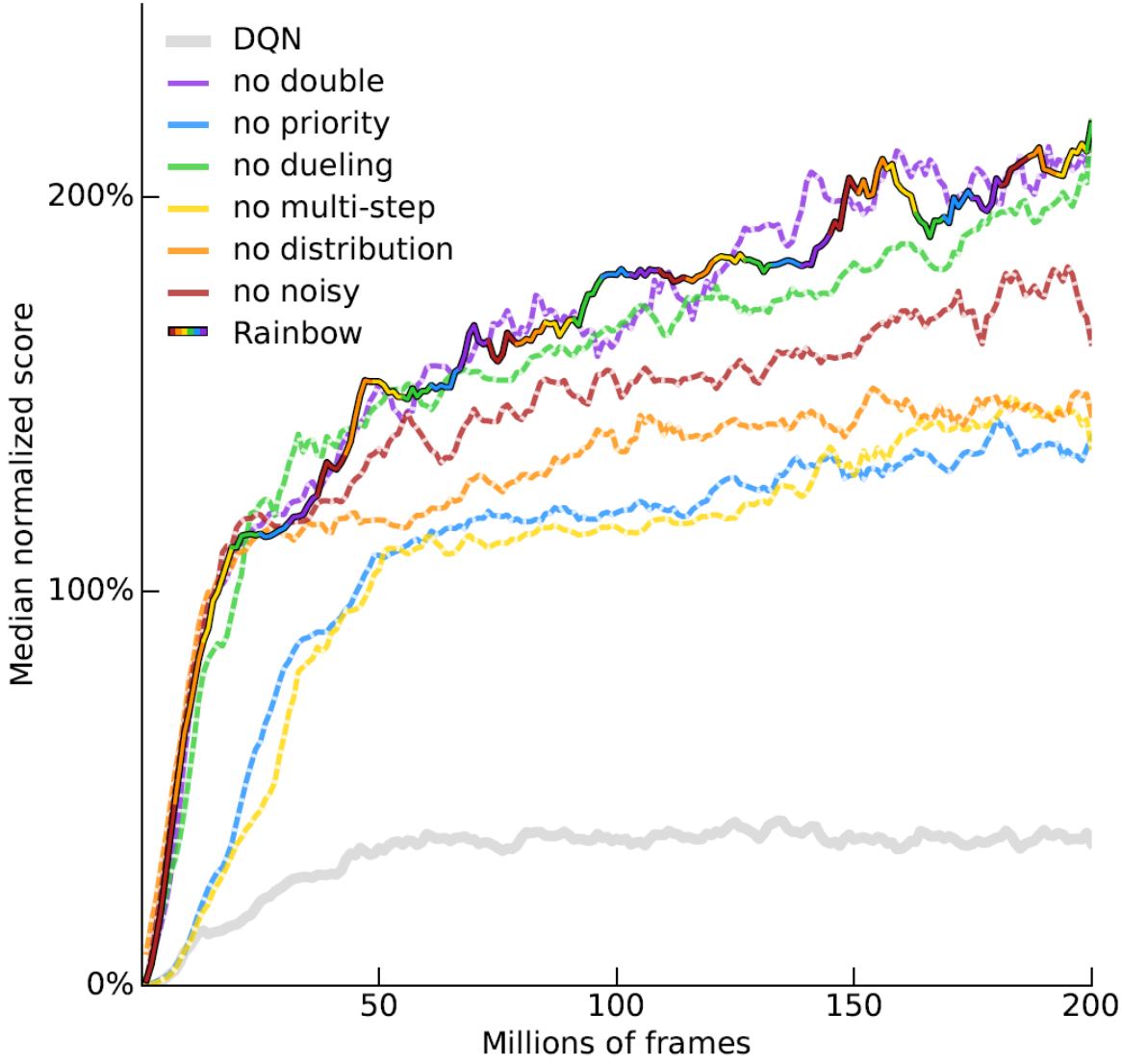
200M frames
≒ 10days by single GPU

Dueling Network

$$p_i(s,a) \propto \exp(V_i(s) + A_i(s,a))$$

Preferred Networks

どの技術が重要なのか？　比較実験(Hessel+, 17)



1つずつ要素技術
を省いたときの性能

Prioritized experience replay
Multi-step learning
Distributional RL
は特に重要そう

Double DQNと
Dueling Networksは
それほど重要でもない？

連続行動への拡張について

- (DDPG) ←次回
- NAF (Gu+16)
- input convex NN (Amos+ 17)

Normalized Advantage Function (NAF) (Gu+16)

連続行動aでもQ関数の最大値を求めたい

$$y_t = r_{t+1} + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta')$$

最大値を求めやすい形式に変更

$$Q(s,a;\theta) = \underbrace{-\frac{1}{2}(a-\mu_\theta(s))^T \Lambda_\theta(s)(a-\mu_\theta(s))}_{\text{アドバンテージ関数}} + V_\theta(s)$$

学習はDQNと同じ

Preferred Networks

Input Convex Neural Network (ICNN) (Amos+17)

連続行動aでもQ関数の最大値を求めたい

$$y_t = r_{t+1} + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta')$$

最大値を求めやすい形式に変更

$$f(y) = z_k \qquad z_{i+1} = g_i(W_i^{(z)} z_i + W_i^{(y)} y + b_i) \qquad (z_0, W_0^{(z)} = 0)$$

gが凸で非減少な活性化関数，$W_i^{(z)}$ が非負のとき
fは入力yに関して凸関数

この凸な関数をQ関数に利用

Preferred Networks

Input Convex Neural Network (ICNN) (Amos+17)

連続行動aでもQ関数の最大値を求めたい

$$y_t = r_{t+1} + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta')$$

最大値を求めやすい形式に変更

$$\begin{aligned}
u_{i+1} &= \tilde{g}_i(\tilde{W}_i u_i + \tilde{b}_i) \\
z_{i+1} &= g_i \left(W_i^{(z)} \left(z_i \circ [W_i^{(zu)} u_i + b_i^{(z)}]_+ \right) + \right. \\
&\qquad \left. W_i^{(y)} \left(y \circ (W_i^{(yu)} u_i + b_i^{(y)}) \right) + W_i^{(u)} u_i + b_i \right) \\
f(x, y; \theta) &= z_k, \ u_0 = x
\end{aligned}$$

入力yに関して凸関数, 最小値は勾配法で求める

学習はDQNと同じ

Preferred Networks

Input Convex Neural Network (ICNN) (Amos+17)

比較実験

Task	DDPG	NAF	ICNN
Ant	1000.00	999.03	**1056.29**
HalfCheetah	2909.77	2575.16	**3822.99**
Hopper	**1501.33**	1100.43	831.00
Humanoid	524.09	**5000.68**	433.38
HumanoidStandup	134265.96	116399.05	**141217.38**
InvDoubPend	**9358.81**	**9359.59**	**9359.41**
InvPend	**1000.00**	**1000.00**	**1000.00**
Reacher	-6.10	-6.31	**-5.08**
Swimmer	49.79	**69.71**	64.89
Walker2d	**1604.18**	1007.25	298.21

まとめ

DQNの基本(Target fixing & Experience Replay)
は学習させるのに重要

推定分散を減らす試み、探索の効率化が盛んに試されている

学習時間はまだまだ膨大。
状態遷移確率を学習して良い
シミュレーション環境を構築する試みも

Preferred Networks